

Lorikeet: An Efficient Multicast Protocol for the Distribution of Multimedia Streams

Justin Viiret

Thesis submitted for the degree of

Doctor of Philosophy

in

Electrical and Electronic Engineering

at

The University of Adelaide

(Faculty of Engineering, Computer and Mathematical Sciences)

School of Electrical and Electronic Engineering



June 4, 2007

Contents

Signed Statement	viii
Acknowledgements	ix
Abstract	xi
1 Introduction	1
1.1 Background	3
1.1.1 Unicast Delivery	4
1.1.2 Multicast Delivery	5
1.1.3 Tree Construction	6
1.1.4 Delivering Multicast Packets	7
1.1.5 Dynamic Trees	8
1.2 Lorikeet	9
1.3 Thesis Structure	12
1.4 Major Research Contributions	14
2 State of the Art	16
2.1 Internet Protocol (IP) Multicast	17
2.2 Small Group Multicast	21
2.3 Application-Level Multicast and Overlay Networks	23
2.4 Topology-Aware Multicast	27

3	The Steiner Tree Problem in Networks	32
3.1	Introduction	32
3.2	The Steiner Tree Problem in Networks	33
3.3	Exact Solutions	36
3.3.1	Spanning Tree Enumeration	36
3.3.2	Dynamic Programming	37
3.4	Reductions	37
3.5	Suboptimal Heuristics	39
3.5.1	Shortest Paths (SP) Heuristic	40
3.5.2	Minimum Spanning Tree (MST) Heuristic	40
3.5.3	Shortest Paths Terminals (SP-T) Heuristic	40
3.5.4	Shortest Paths with Origin (SP-O) Heuristic	41
3.6	Performance Analysis	42
3.6.1	Exact Methods	42
3.6.2	Heuristics	43
3.6.3	Results	44
3.7	The Steiner Tree Problem in Multicast	47
4	The Lorikeet Protocol	49
4.1	A New Multicast Protocol	49
4.2	Design Goals	50
4.2.1	Application Characteristics	50
4.2.2	Environmental Characteristics	51
4.2.3	Requirements	52
4.3	Network Assumptions	56
4.4	Control and Delivery	58
4.5	The Lorikeet Protocol	61
4.5.1	Notation	62
4.5.2	Joining the Tree	63

4.5.3	Leaving the Tree	66
4.5.4	Rearrangement	67
4.5.5	Data Delivery	74
5	Performance Analysis	77
5.1	Introduction	77
5.2	Other Algorithms	78
5.2.1	Source-Join and Greedy algorithms	79
5.2.2	ARIES	79
5.2.3	Delay-Sensitive Greedy (DSG)	81
5.2.4	REUNITE and HBH	81
5.2.5	HBH	87
5.3	Simulation	88
5.3.1	Simulation Experiments	89
5.4	Complexity Analysis	90
5.5	Tree Cost	95
5.5.1	Lorikeet Join and Rearrangement Operations	95
5.5.2	Comparing Lorikeet and Other Algorithms	99
5.5.3	Incremental Deployment	102
5.6	Summary	104
6	Directory Nodes	106
6.1	Motivation	106
6.2	Directory Nodes	109
6.3	Joining the Tree	111
6.3.1	Algorithm	112
6.3.2	Discussion	116
6.4	Results	121
6.5	Conclusions	124

7	Implementation Concerns	126
7.1	Accessing a Lorikeet stream	126
7.2	Implementing Lorikeet	128
7.2.1	Lorikeet Receivers	128
7.2.2	Lorikeet Capable Routers	129
7.2.3	Lorikeet Sources	134
7.3	Systems Issues	134
7.3.1	Load and Capacity	135
7.3.2	Robustness and Failure Recovery	136
7.3.3	Multiple Simultaneous Operations	138
7.3.4	Handoff in Rearrangement	142
7.3.5	Security	144
7.4	Deployment	146
7.4.1	Content Providers	147
7.4.2	Internet Service Providers	148
7.4.3	End Users	149
7.4.4	Placing Capable Routers	149
8	Conclusions and Future Work	151
8.1	Summary	151
8.2	Potential Implementation-Related Research	156
8.3	Potential Protocol Extension	157
8.3.1	Layered Video Delivery	157
8.3.2	Local Recovery	160
8.3.3	Other Further Work	162
	Bibliography	164

Abstract

Internet Protocol multicast has been standardised since the late 1980's, but is yet to be extensively deployed by most Internet Service Providers. Many organisations are not willing to bear the additional router CPU load and memory requirements that multicast entails, and the IP multicast suite of protocols requires deployment on every router spanned by the multicast group to operate. Additionally, these protocols are predominantly designed for the general case of multiple-source, multiple-receiver transmission and can be complex and inefficient to use in simpler scenarios.

Single-source streaming of multimedia on the Internet is rapidly becoming a very popular application, and is predominantly being served by content providers using simultaneous unicast streams. A multicast transmission protocol designed for this application that can operate without requiring a widely deployed IP multicast infrastructure has the potential to save content-providers and network service providers significant amounts of bandwidth. This protocol should provide packet duplication and forwarding capabilities on routers in the network, rather than pushing this functionality to the receivers themselves, requiring them to become part of the multicast infrastructure.

We describe Lorikeet, a new protocol for the multicast distribution of multimedia streams from a single source. This protocol builds its multicast tree from the source, discovering routers that support the protocol in the network and using them to provide branching in the tree. The tree itself is managed in a decentralised fashion, with joining receivers finding parent routers through a limited, recursive search of the tree. On a participating node, information about the tree's structure is limited

to the addresses of that node's children and its path through the tree back to the source. Unlike most other multicast protocols, a new receiver is connected to the tree using its forward path from the source and packets are delivered through the tree via hop-by-hop delivery over unicast connections between nodes. Lorikeet also actively maintains the tree structure using a localised rearrangement algorithm triggered by a topological change in the tree structure. This rearrangement allows the tree to remain efficient in the face of changes to the receiver population, which can change the shape of the tree over time.

Lorikeet is designed to operate with no further protocol support than that provided by existing Internet unicast protocols. It requires none of the standard IP multicast infrastructure, such as Class D group addressing. Its use of unicast connections between nodes allows it to be deployed incrementally on the network, and its behaviour will degrade to simultaneous unicast when no routers that support the protocol are present at all. However, significant performance gains can be achieved even when there are only a few supporting routers present in the network: Lorikeet produces trees with half the cost of a unicast tree when just 10% of routers are Lorikeet-capable. Lorikeet's tree construction and rearrangement algorithms generate multicast trees of comparable total cost to those created by algorithms of considerably higher message complexity, such as those that employ exhaustive searches of the tree during joins.

We develop the Lorikeet protocol from a set of requirements based on its target application and the properties of the current Internet. After describing the protocol's behaviour, we analyse its message complexity and its performance in terms of tree cost. We also analyse several other multicast protocols from the research literature, comparing their performance to that of Lorikeet in both complete deployment and incremental deployment scenarios.